Scalar Field Visualization I

Goal: know what is a scalar field; what are the standard visualization techniques for 2D scalar fields, including color plots and iso-contouring.
What is a Scalar Field?

- An approximation of certain scalar function in space $f(x,y,z)$. 

Image source: blimpyb.com
What is a Scalar Field?

• The approximation of certain scalar function in space $f(x,y,z)$.

• Representation: Most of time, they come in as some scalar values defined on some sample points.
What is a Scalar Field?

- The approximation of certain scalar function in space $f(x,y,z)$.

- Most of time, they come in as some scalar values defined on some sample points.

- **Visualization primitives:**
  - **Geometry:**
    - iso-contours (2D), iso-surfaces (3D),
  - **Attributes:**
    - colors, transparency (3D), 3D textures.
Goal: know how to *design proper transfer functions*, how to produce color plots

Generate 2D color plots
Do we need to construct/extract additional geometry?

NO!

How to generate color plots?
Steps of generating 2D color plots

1. Design color transfer function
2. Color interpolation
To create a color plot, we need to define a proper **Transfer Function** to set **Color** as a function of Scalar Value. The following shows a simple transfer function.

Scalar values \([s_{\text{min}}, s_{\text{max}}]\) \(\rightarrow\) \([0,1]\) \(\rightarrow\) **Colors**
To create a color plot, we need to define a proper *Transfer Function* to set *Color* as a function of Scalar Value. The following shows a simple transfer function.

Scalar values $\rightarrow [0, 1]$

$$t = \frac{S - S_{\text{min}}}{S_{\text{max}} - S_{\text{min}}}$$

(normalization)
To create a color plot, we need to define a proper *Transfer Function* to set Color as a function of Scalar Value. The following shows a simple transfer function.

![Transfer Function Diagram](image)

What colors should they map to?
To create a color plot, we need to define a proper *Transfer Function* to set *Color* as a function of Scalar Value. The following shows a simple transfer function.

What colors should they map to?

Ex. Rainbow color

\[ [0,1] \rightarrow \text{Colors} \]

\[ 0 \rightarrow \text{blue (hue=240)} \]

\[ 1 \rightarrow \text{red (hue=0)} \]
To create a color plot, we need to define a proper **Transfer Function** to set **Color** as a function of **Scalar Value**. The following shows a simple transfer function.

![Diagram showing color transfer function](image)

**Ex. Rainbow color**

What colors should they map to?

- 0 -> blue (hue=240, large)
- 0 < t < 1 ???
- 1 -> red (hue=0, small)
Normalized data value \[ \begin{array}{c} 0 \\ 1 \end{array} \] \quad \text{Hue}_1 \quad \begin{array}{c} 0 \\ 240 \end{array} \\
\quad \quad 240 \times t
Normalized data value | Hue_1 | Hue
---|---|---
0 | 0 | 240
1 | 240 | 240 × t | ???
Normalized data value → Hue_1 → Hue

Normalized data value: 0 → 240 \times t → 0

Hue_1: 1 → 240 \rightarrow 240

Hue: 240 \rightarrow 240 - Hue_1 → 0
To create a color plot, we need to define a proper **Transfer Function** to set **Color** as a function of Scalar Value. The following shows a simple transfer function.

\[
\begin{align*}
\text{Scalar Value} & \quad \text{Color} \\
0 & \quad \text{blue (hue=240)} \\
1 & \quad \text{red (hue=0)} \\
0 \rightarrow \text{blue (hue=240)} \\
t \rightarrow \text{hue} = 240 - 240t \\
1 \rightarrow \text{red (hue=0)}
\end{align*}
\]

What colors should they map to?

Ex. Rainbow color
To create a color plot, we need to define a proper *Transfer Function* to set Color as a function of Scalar Value. The following shows a simple transfer function.

Scalar values -\(\rightarrow [0,1]\) -\(\rightarrow\) Colors

\[
Hue = 240. -240. \frac{S - S_{min}}{S_{max} - S_{min}}
\]

Verify low and high!
How to achieve the following color scale?

Scalar values -> [0,1]

\[ t = \frac{S - S_{\text{min}}}{S_{\text{max}} - S_{\text{min}}} \]

normalization
How to achieve the following color scale?

Scalar values \(\rightarrow [0, 1]\)

- 0 \(\rightarrow\) white
- 1 \(\rightarrow\) full saturated red (hue=0)
How to achieve the following color scale?

Scalar values -> [0, 1]

0 -> white
1 -> full saturated red (hue=0)

Saturation=0
What is between 0 and 1?
Saturation=1
How to achieve the following color scale?

Scalar values $\rightarrow [0,1]$

- 0 $\rightarrow$ white
- 1 $\rightarrow$ full saturated red (hue=0)

$Saturation = t$, where $t$ is in $[0, 1]$
Use the Proper Transfer Function Color Scale to Represent a Range of Scalar Values
Rainbow Scale

\[ \text{Hue} = 240 - 240 \cdot \frac{S - S_{\text{min}}}{S_{\text{max}} - S_{\text{min}}} \]
Gray Scale

How to generate gray scale color scheme?
How to generate gray scale color scheme?

Set r=g=b= the same value (say, the normalized data value between 0 and 1)
Intensity and Saturation Color Scales
Two-Color Interpolation

How to achieve the above color scheme?
Two-Color Interpolation

How to achieve the above color scheme?

You may try a simple linear interpolation of the three color-channels of the two colors.
Heated Object Color Scale

Implementation: add one color component at a time

R+G+B
The diagram illustrates the color spectrum from red to blue, with the color components R, G, and B. The RGB values are shown at different points along the spectrum:

- At the red end: R=1, G=0, B=0
- At the green end: R=0, G=1, B=0
- At the blue end: R=0, G=0, B=1

The spectrum is divided into segments, indicated by the arrows, with the RGB values at each segment:

- [0, 1/3, 2/3, 1]
For $s$ in $[0, 1/3]$, fixed $G=B=0$, map $s$ to $R$ from $[0, 1/3]$ to $[0, 1]$, that is, $R=3.0*s$.
For \( s \) in \([0, 1/3]\), fixed \( G=B=0 \), map \( s \) to \( R \) from \([0, 1/3]\) to \([0, 1]\), that is, \( R=3.0 \cdot s \)

For \( s \) in \((1/3, 2/3]\), fixed \( R=1, B=0 \), map \( s \) to \( G \) from \((1/3, 2/3]\) to \([0, 1]\), that is, \( G=? \)

For \( s \) in \((2/3, 1]\), fixed \( R=G=1 \), map \( s \) to \( B \) from \((2/3, 1]\) to \([0, 1]\), that is, \( B=? \)

Figuring out how to determine \( G \) and \( B \) is part of your assignment 2!
Add-One-Component at a time -- an extension from the heated object color scale

- R+G+B
- R+B+G
- G+R+B
- G+B+R
- B+R+G
- B+G+R
Blue-White-Red Color Scale

Diverging color scheme!
Blue-White-Red Color Scale

How to achieve it?
Blue-White-Red Color Scale

Scalar value -> [0, ½, 1]

How to achieve it?
Blue-White-Red Color Scale

How to achieve it?

Scalar value -> [0, ½, 1]

For [0, ½], saturation **reduces** from 1 to 0, hue=240 (fixed)
For (½, 1], saturation **increases** from 0 to 1, hue=0 (fixed)

Figuring out how to update the saturation is part of your **assignment 2**!
(Discrete) Color Scale Contour

Qualitative/discrete color scheme

Source: http://glscene.sourceforge.net
Sequential color scales can also be discretized to help read the configuration of the scalar fields!

from AccuWeather.com
Use VTK color look-up table (discrete!)

Create and initialize a (color) look up table

```python
lut = vtk.vtkLookupTable() # Initialize the vtk lookup table
cmp = 256 # the size of the table - increase this number to obtain more precise color map
lut.SetNumberOfTableValues(nc)
lut.Build()
```

Generate the individual entries of the look up table (rainbow color is shown below)

```python
sMin = 0.
sMax = 1.

hsv = [0.0, 1.0, 1.0]

for i in range(0, nc):
s = float(i) / nc # uniformly interpolate the scalar values
hsv[0] = 240. - 240. *(s-sMin)/(sMax-sMin) # rainbow color calculation
rgba = hsvRgb(hsv) # convert hsv to rgb (function provided!)
rgba.append(1.0) # set alpha (or opacity) channel
lut.SetTableValue(i, *rgba)
```

**Diagram:**

```
<table>
<thead>
<tr>
<th>entry 0</th>
<th>color 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>entry nc-1</td>
<td>color nc-1</td>
</tr>
</tbody>
</table>
```
Generate 2D color plots

1. Color transfer function
2. Color interpolation
2D Interpolated Color Plots

- How can we turn the **discrete** samples into a **continuous** color plot?
- Here's the **input**: we have a **2D grid** of data points. At each node, we have an X, Y, Z, and a scalar value S. We know $S_{\text{min}}$, $S_{\text{max}}$, and a **Transfer Function**.

Even though this is a 2D technique, we keep around the X, Y, and Z coordinates so that the grid doesn’t have to lie in any particular plane.
2D Interpolated Color Plots

• Let us look at one **square (or quad)** of the mesh at a time

For each scalar value at a vertex

float arrays hsv[3], rgb[3]
hsv[0] = 240. − 240. \frac{S−S_{\text{min}}}{S_{\text{max}}−S_{\text{min}}};
HsvRgb (hsv, rgb);

Convert hsv color to rgb color as monitor uses additive color model!
Use VTK color look-up table with a mapper

Get scalar value range

\[
\text{min\_scalar, max\_scalar} = \\
vtk\_geometry.\text{GetOutput().GetPointData().GetArray("s").GetRange()}
\]

Create a vtkPolyDataMapper object and connect it to the geometry of the data domain (i.e., the grid). Assume a look up table, \text{lut}, has been built.

\[
\begin{align*}
\text{vtk\_poly\_mapper} &= \text{vtk\_.\text{vtkPolyDataMapper()}} \\
\text{vtk\_poly\_mapper}\text{.SetInputData(vtk\_geometry.GetOutput())}
\end{align*}
\]

\[
\begin{align*}
\text{vtk\_poly\_mapper}\text{.SetScalarModeToUsePointData()}
\text{vtk\_poly\_mapper}\text{.SetLookupTable(lut)}
\text{vtk\_poly\_mapper}\text{.SetScalarRange(min\_scalar, max\_scalar)} \#\text{Specify range in terms of scalar minimum and maximum (smin,smax). These values are used to map scalars into lookup table}
\text{vtk\_poly\_mapper}\text{.SelectColorArray(scalar\_field)} \#\text{set the name of scalar field}
\end{align*}
\]

\[
\begin{array}{c}
s\text{Min} \rightarrow \text{entry 0} \\
\hline
\text{color 0} \\
\hline
\ldots
\hline
\text{color nc-1} \\
\hline
s\text{Max} \rightarrow \text{entry nc-1}
\end{array}
\]
2D Interpolated Color Plots

- What happen underneath is the calling of OpenGL drawing functions

Loop through the individual quads to perform the following for each quad

```c
// compute color at V0
glColor3fv (rgb0);
glVertex3f (x0, y0, z0);

// compute color at V1
glColor3fv (rgb1);
glVertex3f (x1, y1, z1);

// compute color at V3
glColor3fv (rgb3);
glVertex3f (x3, y3, z3);

// compute color at V2
glColor3fv (rgb2);
glVertex3f (x2, y2, z2);
```
If the grid is unstructured like a triangle mesh, then...

Loop through the individual triangles to perform the following for each triangle

```c
// compute color at V0
setColor3fv (rgb0);
setPosition3f (x0, y0, z0);

// compute color at V1
setColor3fv (rgb1);
setPosition3f (x1, y1, z1);

// compute color at V3
setColor3fv (rgb3);
setPosition3f (x3, y3, z3);
```

OpenGL mechanism